INTEGRATING DIS AND SIMNET INTO HLA WITH A GATEWAY

Andy Cox, Douglas D. Wood, Mikel D. Petty, and Kenneth A. Juge
Institute for Simulation and Training
3280 Progress Drive
Orlando, FL 32826

KEYWORDS
HLA, ProtoFederation, Legacy Integration, Gateway, Computer Generated Forces

ABSTRACT

The High Level Architecture (HLA) is a project to develop a simulation infrastructure that will promote interoperability between simulations. The Defense Modeling and Simulation Office (DMSO) commissioned several experimental applications of HLA in 1996 to test and refine the HLA concept. One of those experiments was conducted by the Platform Proto-Federation (PPF), a group of virtual real-time (i.e., DIS-type) simulations assembled to test HLA in that domain. The PPF consisted of four member programs: BDS-D, BFTT, CCTT, and JTCTS. The BDS-D federate was a crewed SIMNET M1 simulator. BDS-D connected the M1 simulator to the HLA network using an HLA Gateway, developed at IST. The HLA Gateway performs protocol translation, converting SIMNET PDUs into RTI service invocations, and vice versa. The Gateway approach was chosen for BDS-D to test the feasibility of integrating legacy simulations into HLA in that manner. In a parallel effort, IST added DIS functionality to the Gateway, allowing it to perform the same translation for DIS. This paper will describe the design and implementation of the IST HLA Gateway, a description of the Gateway's Simulation Object Model (SOM), performance of the Gateway in PPF experiments, and analysis of the feasibility of the Gateway as a means of legacy system integration.

1.0 INTRODUCTION

The Defense Modeling and Simulation Office (DMSO) is supporting several experimental applications in 1996 to test and refine the HLA concept. One of those experiments is being conducted by the Platform Proto-Federation (PPF), a group of virtual real-time (i.e., DIS-type) simulations formed to test the applicability of HLA to that particular simulation category. One member of the PPF is the US Army Simulation, Training and Instrumentation Command's Battlefield Distributed Simulation-Developmental (BDS-D) program. The Institute for Simulation and Training (IST) and STRICOM have chosen to connect the BDS-D M1 tank simulator to the larger HLA PPF via an interface node. The interface will convert HLA Runtime Infrastructure (RTI) services into SIMNET PDUs and forward them to the M1 Simulator. Similarly, SIMNET PDUs are translated into RTI service calls. This device is referred to as the IST HLA Gateway. IST and STRICOM have chosen to use the gateway approach to HLA integration so as to test its feasibility. This provides a mechanism for legacy applications, such as the BDS-D M1 crewed simulator, to utilize RTI services without requiring any modifications to existing software. If viable, the interface will enable the integration of the US Army's large inventory of existing SIMNET equipment into HLA exercises.

2.0 IST HLA GATEWAY

Developed on a Silicon Graphics Indy using Irix 5.3, the HLA Gateway translates from SIMNET 6.6.1 or DIS 2.0.3 to HLA and vice versa. Incoming PDUs are converted to attribute updates, with an incoming PDU’s fields compared to attribute values for the object retained in the Gateway, so that only changed attributes are updated. Incoming attribute reflections from the RTI are converted to PDUs, with the object’s stored attribute values used to fill in attributes needed for the PDU that are not provided in the reflection. Dead Reckoning models of remote objects are maintained in the Gateway and used to generate heartbeat PDUs when the attribute reflection interarrival time exceeds the protocol’s time-out limit. The Gateway performs data transformation (e.g. coordinate conversion) to convert the protocols’ PDU fields into the attributes. Interactions are also converted into DIS/SIMNET PDUs, again performing any needed data conversion in the process.

2.1 Capabilities

The Gateway used to link the SIMNET M1 simulator to the HLA network is in fact a multi-functional HLA node with a range of useful capabilities. In short, they are:

1. HLA-SIMNET protocol translation
2. HLA-DIS protocol translation
3. HLA Log and Playback
4. Computer Generated Forces
5. Run-time Object Monitor

All of these capabilities were designed into a single software module because of the significant amount of shared code and functionality between them. Gateway design is discussed in greater detail in section 2.2.

2.1.1 Protocol Translation

Currently the HLA Gateway translates the most commonly used PDU types in each protocol: Vehicle Appearance, Fire, Indirect Fire, Impact, and Collision for SIMNET and Entity State, Fire, Detonation, and Collision for DIS. These were sufficient to support the M1 simulator’s role as prescribed in the PPF experiment plan. The HLA Gateway’s publications and subscriptions, which are set in a configuration file and can be changed at run-time, are based on the PPF FOM, which is approximately a subset of DIS. Presently, the HLA Gateway does not handle HLA ownership transfer functions. Additional RTI services such as query, delete, and time management will be considered to increase HLA functionality. Several other enhancements are anticipated, including upgrading the Gateway’s HLA communication to a DIS “Super-FOM” (a statement of the DIS standard in OMT format), adding translation for additional SIMNET and DIS PDUs, and investigating the DIS Data Dictionary/Protocol Catalog to facilitate standardization of class names, data types, and enumerations.

2.1.2. HLA Log and Playback

The HLA Gateway can log incoming HLA attribute reflections and interactions to data files in both text and binary form. The text log files are useful for analysis and debugging. The binary log files can be played back by the Gateway; during a playback, the Gateway sends the sequence of attribute reflections and interactions recorded in the log file. Replay is based on receive time, though a replay based on send time is being considered. The log and playback can be controlled by “Start”, “Pause”, “Resume”, or “Stop” commands.

The Log function works by subscription, i.e., it will log everything that the HLA Gateway is subscribed to. There are, of course, maximum recording rates and amounts. However, the logging capabilities are not constrained by current HLA implementations. Some HLA events are missed with a subscription-based approach to logging (e.g. Federation Joins), but this capability has nevertheless been very useful for experimentation and debugging.

2.1.3 Computer Generated Forces

Integrated into the HLA Gateway is a UNIX version of IST's Computer Generated Forces system, which has been developed and extended since 1990. The Gateway's CGF component can generate and control entities in a SIMNET, DIS, or HLA exercise. While running as an HLA-based CGF, the Gateway communicates directly in HLA, i.e., no DIS or SIMNET PDUs are generated. The CGF component provides a Plan View Display, showing an overhead map view of the exercise. The Gateway's CGF was a valuable tool during PPF testing as a means to quickly instantiate objects in a trial.

The CGF system is limited to about 40 locally owned objects and is operated by a command interface. It currently operates in a real-time independent time advance mode and does not accommodate any of the HLA time management services.

2.1.4 Run-time Object Monitor

The Run-time Object Monitor is a run-time command driven facility that allows the operator to monitor the status of HLA objects during the execution of an HLA exercise. In particular, the attributes of any selected object can be displayed, or a summary of all objects can be generated.

2.2 Gateway Design

The BDS-D/HLA Gateway is an extension of IST's Computer Generated Forces (CGF) Testbed (Smith, 1992). The Testbed provided a SIMNET and DIS compliant framework for coordinate conversions, dead reckoning, visual displays, and a protocol-independent simulation core. Gateway functionality entailed the addition of an RTI interface and set of services in accordance with the HLA Specification.

2.2.1 Component Services Framework

The Gateway's RTI interface was developed using TASC's Component Services Framework (CSF), a set of middleware that encapsulates and extends RTI services in C++ classes (Bachinsky, 1996). To support translation between networks, two network interfaces are supported within the gateway. The use of separate networks has several benefits, including reduced processing at the simulation hosts and a lower average utilization on each network. A byproduct of the gateway's translation is that the entire HLA exercise may be observed remotely on the SIMNET network by other passive monitoring devices such as Plan View or Stealth visual displays. A diagram of the gateway and the overall PPF network is shown in Figure 1.

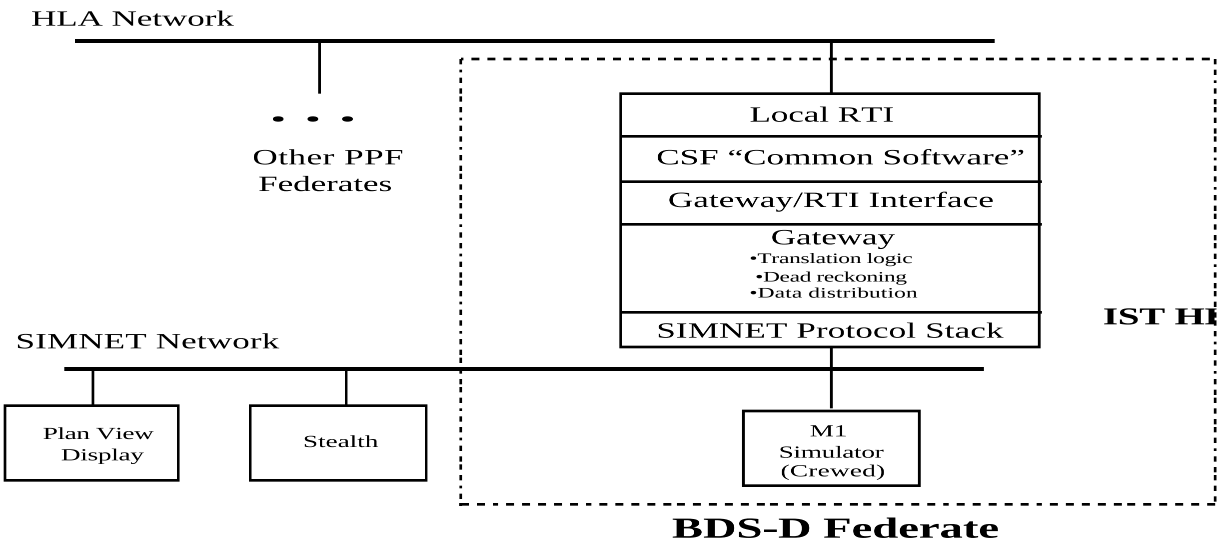

Figure 1. IST HLA Gateway and PPF Network

The Gateway comprises several "manager" processes, each providing a specific type of service. The major processes used in the Gateway are listed in Table 1. The Protocol Manager was the only existing manager significantly effected by the addition of the gateway functionality, other than additional console input processed by the Console Manager. The Protocol Manager provides an interface between the internal application and the external protocols. The Protocol Manager receives incoming data from the simulation network and performs any necessary data transformations to create an internal representation. Similarly, outgoing data from the application is transformed into the appropriate simulation protocol format. Incoming data is forwarded to the Distribution Manager via the Executive message queue for further processing (i.e., dead reckoning model update). An additional communication path was added to the Protocol Manager to pass the data to the Gateway Manager via a direct function call. Output to the legacy simulation network via direct function calls already existed.

Similar to the Protocol Manager, the Gateway Manager provides an interface between the internal application and the HLA RTI. The Gateway Manager receives incoming data from an RTI Interface (see next section) and performs data transformations. For interactions, there is practically no difference; an HLA RTI interaction is always a complete set of data. In contrast, HLA RTI object updates or reflections are mostly partial data sets containing only changed data, excluding the first update or instantiation where a complete update is expected. Upon receiving an object update from the RTI Interface, the Gateway Manager gets a current copy of the entity's dead reckoned model from the Distribution Manager or creates one if this is the first update. The partial updates are applied to this state information which is then forwarded back to the Distribution Manager and simultaneously passed to the Protocol Manager for distribution to the legacy simulation network. Interactions are similarly distributed to both of these Managers.

Manager	Function
Console Manager	Process console input
Display Manager	PVD/3D Display
Distribution Manager	Distribute information from the network
Executive	Message Scheduler
Gateway Manager	Provide RTI and Gateway services
Initialization Manager	Initialization and removal of entities
Protocol Manager	Provide protocol specific interfaces
Radio Manager	DIS Radio Handler
Simulation Manager	DIS Simulation Management services

Table 1. Gateway Manager Processes

The Gateway Manager accepts direct function calls to send application data to the HLA RTI via the RTI Interface, first performing any appropriate data transformations. A state diagram depicting the communication between the Protocol Manager, Distribution Manager, and Gateway Manager is given in Figure 2. The solid lines are messages or incoming data and the dotted lines are direct function calls.

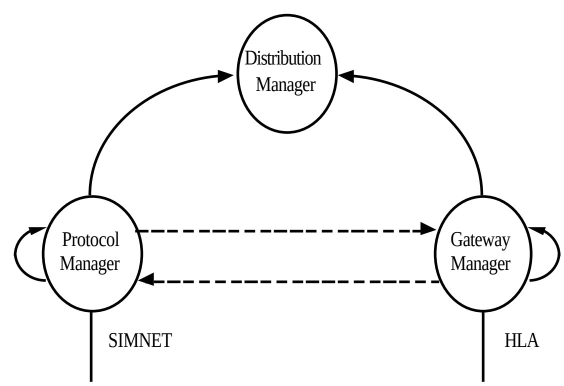

Figure 2. Gateway Message Flow

The majority of the Gateway's modifications were related to the exchange of data between the legacy simulation protocol and the HLA RTI. However, the Gateway must provide additional functionality to meet the HLA Interface Specification requirements as implemented by the HLA RTI. The HLA RTI requires an application or federate to join a federation and to publish and subscribe to object and interaction classes. The federate must also be able to respond to calls invoked by the HLA RTI on the federate (i.e., incoming HLA data). The implementation of these functions and the transformation of application data into HLA RTI calls is described below.

2.2.2 RTI Interface

The Gateway's RTI Interface is implemented through the four simple API functions shown in Table 2.

API Service	Function
Start	Initialization Services
Stop	Termination Services
Send	Send Data to RTI
Receive	Process Incoming RTI data

Table 2. Gateway RTI Services

The *start* function initializes the RTI Interface components and using these components it invokes the RTI services to create a federation (optional), join a federation, and subscribe and publish to object and interaction classes. The RTI Interface components are parameterized via a configuration file that contains parameters for naming the federation, specifying appropriate host machines (executive and ambassador), selecting subscription and publication classes, and determining whether to create a federation or join an existing one. As was mentioned above, the RTI Interface was developed on top of the CSF. The CSF components that are created during initialization are given in Table 3.

The RTI Interface defines a publisher and a subscriber component to support the use of the CSF managers. The publisher and subscriber components define what object and interactions classes are published and subscribed (i.e., classes of data to be output and received). They are also used to support the transformation of data from the application into calls on the CSF Managers and vice versa. These two components are initialized and used for publication and subscription when start is called.

CSF Manager	Functions
Exercise	federation creation / join / resign / destroy
FOM	maintain transient database of object classes and their published attributes and interactions
Interest	subscription, database buffering of object reflections and interaction generations, process object and interaction queries
Object	publication, create objects, update object attributes, send interaction
Process	abstraction for HLA network event processing

Table 3. CSF Managers

The RTI Interface *send* function accepts the full description of an entity state or entity interaction and transforms the data into object attributes and interaction parameters for delivery to the CSF Object Manager (OM). Since interaction parameters are not named, all interaction attributes are required and their positioned order is fixed. In contrast, object attribute updates are named and should only be sent when a change has occurred. Because the CSF OM determines when an attribute has changed, the RTI Interface delivers the complete set of entity state information to the CSF OM. The CSF OM only sends out those attributes that have changed. The design of the CSF OM also supports filtering attribute updates for those attributes not activated by the RTI (i.e., has no subscribers); however, this RTI capability has not been implemented.

The RTI Interface *receive* function first invokes the CSF Process Manager (PM) to process the RTI event queue (e.g., object reflection and interaction generation) and fill the CSF Interest Manager's (IM) object attribute and interaction database. Queries are then invoked on the CSF IM's database to extract the updated attributes and interactions. A message is constructed for each set of object attributes and for each interaction. The messages are sent to the Gateway Manager for data transformation and delivery to the Protocol Manager and the Distribution Manager. The database is then cleared for the next processing cycle.

The RTI Interface *stop* function invokes the CSF Exercise manger to resign from the federation and optionally destroy it.

Figure 3 shows the communication between the Gateway Manager and the RTI Interface.

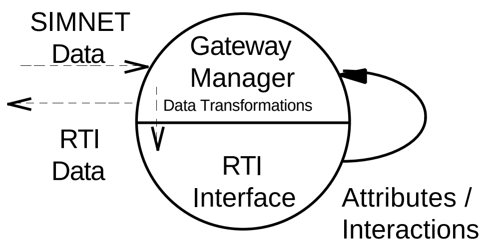

Figure 3. Gateway Manager and RTI Interface Communication

It is significant to note that the specification of class and attribute names and attribute and parameter data types in the FOM is ingrained in the Gateway source code. Changes to the FOM would require recoding and recompiling the Gateway. However, these changes were mitigated by restricting affected code to the RTI Interface. To a small extent the design of the RTI Interface attempted to provide the flexibility to adapt to changing FOMs. The attributes and parameters used for exchanging data between the RTI Interface and the CSF (or RTI) are defined as “any” types or streams. These streams can store any type of data and are tagged with the data type they contain. The RTI Interface extends these types by adding two additional tags, one to identify the type used internally and another to identify the type specified in the FOM. These additional tags can be used to perform data transformations when extracting data from a stream or putting data into a stream. For example, this technique allows a LOCATION attribute to be defined as three floats internally and three doubles in the FOM. Additional steps could be made to allow class and attribute names to be changed and initialized at startup via configuration files.

2.2.3 Simulation Object Model

The HLA Specification defines the Simulation Object Model (SOM) to prescribe the specific types and format of data of an individual simulation. (Object Model Template, 1996). The BDS-D SOM documents the key information about the BDS-D federate, including its objects, attributes, associations, and interactions. A copy of the BDS-D M1 SOM is not included due to length restrictions.

The model used to begin work on a BDS-D SOM was based on the information included in SIMNET BBN Report No. 6787, which was slightly out of date. Therefore, the first task was to update it and expand it, modeling after the HLA-PPF FOM which was then available. SIMNET BBN Report No. 7627, a more recent description of the SIMNET network and protocols, was used, as well as header files from the SIMNET application itself.

An initial version was developed which conformed to the then most recent version of the Object Model Template (OMT), version 0.2. This version was as streamlined as possible, and dealt with a single entity, the M1. This minimal version was released April 23, 1996.

One of the issues that arose during the development of the initial SOM was that of whether the SOM should describe only those classes, interactions, attributes that the Federate could publish, or those that it could publish and subscribe. Therefore, a second version of the SOM was built which contained a column “Publish/Subscribe”, and more material was added to the SOM. This material was the objects, attributes, and interactions to which the BDS-D could only subscribe. This made for a larger, more complete SOM.

An issue that arose at this point was that of representing complex datatypes. A complex datatypes table was added to the SOM, detailing both complex datatypes and enumerations. The inclusion of this table allowed for more information to be included without cluttering the attribute table, thus making it more readable.

Having the datatypes from both SIMNET and HLA in a single SOM made the document cluttered and difficult to follow. As a result, it was decided to split the SOM into two parts, a SIMNET SOM and an HLA SOM. These two parts represent the two sides of our gateway, with separate interfaces to the SIMNET and HLA networks. Breaking up the SOM has several advantages:

- multiple representations of datatypes are no longer a problem
- Gateway functionality is more clearly documented for the purposes of constructing a FOM for a new federation out of multiple SOM’s
- it is possible to determine what parts of SIMNET the gateway can support, so if the gateway is ported to a different SIMNET simulator, the needed changes to the gateway will be apparent from looking at the SIMNET SOM
- having the two SOM’s will aid in the construction and maintenance of the gateway, since they document what is expected from the

gateway in the way of conversions between SIMNET and HLA.

2.3 Design Issues

Some of the various issues addressed during Gateway development are discussed in this section, including communications services, the issue of dead reckoning, the DIS/SIMNET "heartbeat" mechanism, and the use of object and entity identifiers in the RTI and DIS.

2.3.1 Communication Services

There are significant differences between the communication services in SIMNET, DIS, and the RTI. SIMNET defines an association protocol (AP) to provide communication services underlying the simulation and data collection protocols (Pope, 1991). For HLA applications, communication is provided entirely by the RTI using services defined in the Interface Specification. Each protocol is used to provide both reliable and unreliable services, although in many cases the gateway's translation requires a compromise between incompatible requirements. For example, some SIMNET PDUs may be issued via a transaction service of the association protocol. The RTI, however, may be operating in a best effort mode using an unreliable broadcast or multicast datagram service in which no acknowledgment is possible. A similar incompatibility will occur if the RTI is providing reliable service for communications utilizing datagram transmission in SIMNET or DIS. Resolutions to issues such as these are frequently deficient in some regard.

2.3.2 Dead Reckoning and Periodic Updates

Dead Reckoning and Periodic Updates are two factors that significantly influenced gateway design. Dead Reckoning is a technique that reduces the frequency at which information must be transmitted via the underlying network. In addition to high fidelity dynamics information, each simulator maintains a dead reckoning model of itself and of all remote entities.

The DR model is used to extrapolate Time, Space, and Position Information (TSPI) to depict remote entities in the interval between updates. A simulator compares its high fidelity information with its DR model, and updates are issued only if some threshold value has been exceeded. Periodic updates, in contrast, often result in the transmission of redundant information. Regular updates provide several positive benefits, however, such as mitigating the impact of data loss in an unreliable network, allowing entities to quickly capture the state of an exercise regardless of the time of entry, and in regulating the amount of error that can be introduced by Dead Reckoning. The RTI is expected to provide minimum rate and state consistent categories of service to support these considerations.

DIS entities issue an Entity State PDU based on one of three conditions: a change in discrete appearance attributes, a change in TSPI that exceeds some DR threshold, or upon the expiration of a timer. Similarly, remote entities are considered to have timed out and are removed if a prescribed time has elapsed without receipt of an Entity State PDU. In most categories of service, the RTI transmits only changes in state information, and does not transmit periodic updates. Accordingly, HLA entities do not time out, and must be explicitly removed. The gateway must ensure that periodic updates are provided to the DIS network. Two approaches are possible:

1. The gateway uses RTI services to query state information on a regular basis.
2. The gateway maintains state information on each HLA entity.

The first option was dismissed as requiring substantial overhead given the frequency at which the DIS network must be updated. The second option requires that the gateway maintain a dead reckoning model for each remote HLA entity to which it is subscribed. Given a DR model for each HLA entity, the gateway can determine with minimal overhead when a DIS PDU should be issued in the absence of an HLA update.

All attributes and interactions received from the RTI result in the transmission of a DIS PDU. Incoming DIS data, however, does not always require the invocation of an RTI service. Only changes in state information are required to be transmitted to the RTI. In the case of a quiescent DIS entity, no update is required. As previously stated, there are three conditions that require the issue of a DIS PDU. One of these includes a special case that led to considerable discussion. A moving DIS entity can conceivably be within its DR thresholds at the expiration of the heartbeat interval. At that time, it will issue an Appearance PDU. The location within the PDU will have changed. If this information is not forwarded to the RTI, then error will be introduced between the DIS and RTI DR representations of the issuing entity. This could possibly be addressed by maintaining two DR models within the gateway. However, an equally plausible resolution is to forward the new TSPI to the RTI, while still filtering out any unchanged data. This reduces the computational overhead within the gateway and should be a more scaleable solution.

2.3.3 Object ID versus Entity ID

DIS identified objects through a Site/Application/Entity ID, each component consisting of two octets. RTI objects are assigned an object identifier of four octets. The DIS ID has additional meaning beyond simple object identification, providing logical classifications based on geographic location (site) and host workstation, if applicable. If this additional information is not required, an RTI object ID can be used to generate a unique DIS ID. One approach is to use the high order octet as the DIS site, the next octet as the host, and the two low order octets as the entity ID. For example, object ID 0x01234567 would map to the unique DIS ID site 0x0001, host 0x0023, entity 0x4567. The HLA-PPF FOM included a Site/Host/Entity ID as a FOM attribute, in addition to the RTI Object ID, and the owning application explicitly assigned an Entity ID in addition to the RTI-assigned Object ID.

2.3.4 Exercise ID versus Federation Name

An exercise ID and federation name are used in DIS and by the RTI, respectively, as a distinct exercise identifier. These parameters are configured prior to run time within the gateway. Currently, the Gateway supports a single DIS exercise at one time. However, a Federation Name to Exercise ID mapping could be accomplished with minimal effort.

3.0 LEGACY INTEGRATION

One of the goals of IST's participation in the PPF experiment was to determine the feasibility of integrating legacy systems into an HLA federation using a gateway approach. We believe that the success of the BDS-D HLA Gateway in the PPF experiment demonstrates that feasibility, at least for SIMNET and DIS simulations. A range of SIMNET devices (M1 Simulator, Stealth display, and Plan View Display, and PC-based Computer Generated Forces) were active parts of the PPF's trials via the Gateway, and their integration was achieved with no software or hardware modifications to the SIMNET devices at all. The DIS protocol translation built into the Gateway makes similar results achievable for DIS simulators.

Clearly, HLA integration using a Gateway adds data communications latency to perform the protocol translation; that is unavoidable. However, the run-time performance of the BDS-D HLA Gateway, while certainly needing attention, was encouragingly good, especially given its status as a prototype. The Gateway's performance is certain to improve, due both to ongoing optimization within the Gateway and planned performance-related enhancements to the RTI.

4.0 CONCLUSIONS

The gateway has been successful in providing a link between SIMNET applications and HLA-based simulations exchanging data through the Run Time Infrastructure. The use of a stand alone internetworking device appears to be a viable approach for cases in which re-engineering costs to achieve HLA interoperability may be prohibitive. We conclude that integrating legacy systems, in particular SIMNET and DIS simulations, into HLA federations with a Gateway is a feasible, convenient, and cost effective alternative.

Preliminary analysis of the data collected during PPF experiments have shown end to end latencies for RTI communication are significantly higher than similar communication via standard DIS implementations. For example, two applications exchanging appearance updates on a local Ethernet network under light load frequently exhibit greater than 100 milliseconds latency using RTI best effort (broadcast) communication, whereas DIS packets typically incur from 1-2 milliseconds under similar conditions. The 100ms does not include processing within the IST HLA Gateway, which required an additional 1 to 20 milliseconds to process each RTI message, depending on the number and type of attributes. It is anticipated that Gateway performance may be increased through optimizations in the Gateway, CSF, and RTI.

Detailed measurements of Gateway and RTI performance are still underway at the time of this report. Final results of the analysis of PPF experiments are scheduled to be complete in early September 1996.

5.0 REFERENCES

Bachinsky, S. T., Hancock, J. P., Hooks, M. L., & Rybacki, R. M. (1996) Common Software Delivery 1: Status, Plan, Design, and API. TASC.

Harkrider, S. M. and Petty, M. D. (1996). "High Level Architecture and the Platform Proto-Federation", Proceedings of the 18th Interservice/Industry Training Systems and Education Conference, Orlando FL, December 3-6 1996.

Pope, A. R. The SIMNET Network and Protocols. (1991) Version 6.6.1. BBN.

Smith, S. H., Karr, C. R., Petty, M. D., Franceschini, R. W., & Watkins, J. E. (1992) "The IST Semi-Automated Forces Testbed." IST.

Department of Defense. (1996) "High Level Architecture for Simulations, Object Model Template Draft v0.2." DMSO.

Department of Defense. (1996) "High Level Architecture for Simulations, Interface Specification Draft v0.5." DMSO.

6.0 ABOUT THE AUTHORS

Andy Cox is an Associate Computer Scientist at the Institute for Simulation and Training. He is currently involved in various projects in the areas of Distributed Interactive Simulation and the High Level Architecture. Mr. Cox received a Bachelor of Science degree in Computer Science from the University of Central Florida and is currently pursuing a graduate degree. His background includes prior military service as an Infantryman, Quartermaster Officer, and Military Police Officer. His research interests are in internetworking and distributed simulation.

Douglas D. Wood is a Research Computer Scientist at the Institute for Simulation and Training. Mr. Wood has performed research primarily in the area of distributed simulation, including HLA experiments, electronic warfare protocols, and algorithms for computer generated forces. He has also performed research in simulation for emergency management training. Mr. Wood received a M.S. and a B.S. in Computer Science from the University of Central Florida.

Mikel D. Petty is a Program Manager and Senior Research Computer Scientist at the Institute for Simulation and Training. He is currently leading IST's HLA BDS-D project; previously he managed IST s Emergency Management and Computer Generated Forces research. Mr. Petty received a B.S. in Computer Science from the California State University Sacramento and a M.S. in Computer Science from the University of Central Florida, and is a Ph.D. student in Computer Science at UCF. His research interests are in simulation and computational geometry.

Kenneth A. Juge is a Research Assistant at the Institute for Simulation and Training (IST), working on the High Level Architecture BDS-D project. Mr. Juge received a B.A. in Physics from New College and a M.S. in Physics from Michigan State University. He is currently a Ph.D. student in Mechanical Engineering at the University of Central Florida.